Lab 1

Jordan Small

October 1, 2023

CDA3203 Computer Logic Design

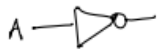
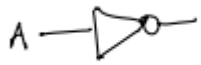
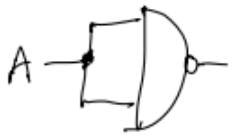
Fall 2023

Dr. Maria Petrie

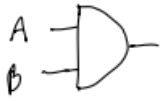
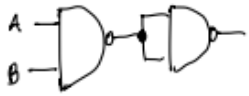
Florida Atlantic University

Part 1: Design 9 circuits by completing below: Draw the symbol for the gate, its Truth Table, its Simplest Sum of Products Expression, draw its NOT-AND-OR Equivalent Circuit, its all-NAND Equivalent Circuit.

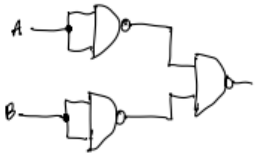
1.1 NOT gate.

<table>
<tr><th>Draw NOT gate</th><th>Truth Table and Simplest Sum of Products Equation</th><th>NOT-AND-OR Equivalent Circuit</th><th>all-NAND Equivalent Circuit</th></tr>
<tr>
<td> 
</td>
<td>
<table>
<tr><th>A</th><th>NOT(A)</th></tr>
<tr><td>1</td><td>0</td></tr>
<tr><td>0</td><td>1</td></tr>
</table>
Y1 = A'
</td>
<td> 
</td>
<td> 
</td>
</tr>
</table>

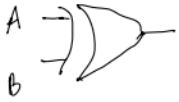
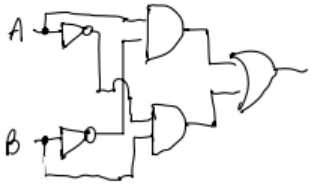
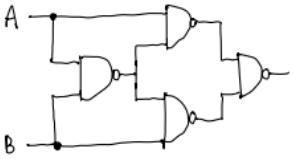
1.2- AND gate

<table>
<tr><th>Draw AND gate</th><th>Truth Table and Simplest Sum of Products Equation</th><th>NOT-AND-OR Equivalent Circuit</th><th>all-NAND Equivalent Circuit</th></tr>
<tr>
<td> 
</td>
<td>
<table>
<tr><th>A</th><th>B</th><th>AND(A,B)</th></tr>
<tr><td>0</td><td>0</td><td>0</td></tr>
<tr><td>0</td><td>1</td><td>0</td></tr>
<tr><td>1</td><td>0</td><td>0</td></tr>
<tr><td>1</td><td>1</td><td>1</td></tr>
</table>
Y2 = AB
</td>
<td> 
</td>
<td> 
</td>
</tr>
</table>

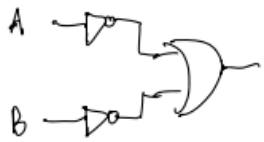
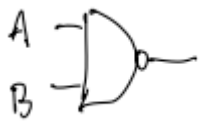
1.3- OR gate

<table>
<tr><th>Draw OR gate</th><th>Truth Table and Simplest Sum of Products Equation</th><th>NOT-AND-OR Equivalent Circuit</th><th>all-NAND Equivalent Circuit</th></tr>
<tr>
<td> 
</td>
<td>
<table>
<tr><th>A</th><th>B</th><th>OR(A,B)</th></tr>
<tr><td>0</td><td>0</td><td>0</td></tr>
<tr><td>0</td><td>1</td><td>1</td></tr>
<tr><td>1</td><td>0</td><td>1</td></tr>
<tr><td>1</td><td>1</td><td>1</td></tr>
</table>
Y3 = A + B
</td>
<td> 
</td>
<td> 
</td>
</tr>
</table>

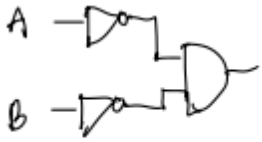
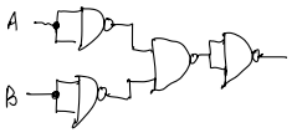
1.4- XOR gate

<table>
<tr><th>Draw XOR gate</th><th>Truth Table and Simplest Sum of Products Equation</th><th>NOT-AND-OR Equivalent Circuit</th><th>all-NAND Equivalent Circuit</th></tr>
<tr>
<td> 
</td>
<td>
<table>
<tr><th>A</th><th>B</th><th>XOR(A,B)</th></tr>
<tr><td>0</td><td>0</td><td>0</td></tr>
<tr><td>0</td><td>1</td><td>1</td></tr>
<tr><td>1</td><td>0</td><td>1</td></tr>
<tr><td>1</td><td>1</td><td>0</td></tr>
</table>
Y4 = AB' + A'B
</td>
<td> 
</td>
<td> 
</td>
</tr>
</table>

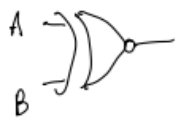
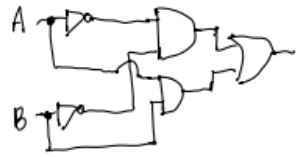
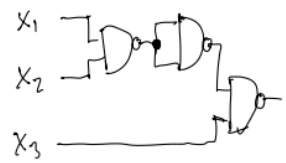
1.5- NAND gate

<table>
<tr><th>Draw NAND gate</th><th>Truth Table and Simplest Sum of Products Equation</th><th>NOT-AND-OR Equivalent Circuit</th><th>all-NAND Equivalent Circuit</th></tr>
<tr>
<td> 
</td>
<td>
<table>
<tr><th>A</th><th>B</th><th>NAND(A,B)</th></tr>
<tr><td>0</td><td>0</td><td>1</td></tr>
<tr><td>0</td><td>1</td><td>1</td></tr>
<tr><td>1</td><td>0</td><td>1</td></tr>
<tr><td>1</td><td>1</td><td>0</td></tr>
</table>
Y5 = A' + B'
</td>
<td> 
</td>
<td> 
</td>
</tr>
</table>

1.6- NOR gate

<table>
<tr><th>Draw NOR gate</th><th>Truth Table and Simplest Sum of Products Equation</th><th>NOT-AND-OR Equivalent Circuit</th><th>all-NAND Equivalent Circuit</th></tr>
<tr>
<td> 
</td>
<td>
<table>
<tr><th>A</th><th>B</th><th>NOR(A,B)</th></tr>
<tr><td>0</td><td>0</td><td>1</td></tr>
<tr><td>0</td><td>1</td><td>0</td></tr>
<tr><td>1</td><td>0</td><td>0</td></tr>
<tr><td>1</td><td>1</td><td>0</td></tr>
</table>
Y6 = A'B'
</td>
<td> 
</td>
<td> 
</td>
</tr>
</table>

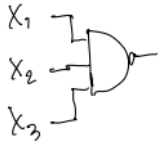
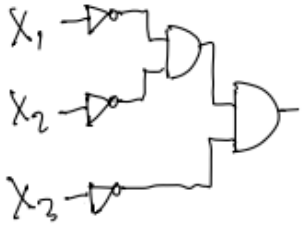
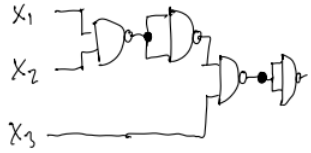
1.7- XNOR gate

Draw XNOR gate	Truth Table and Simplest Sum of Products Equation	NOT-AND-OR Equivalent Circuit	all-NAND Equivalent Circuit
			

A	B	XNOR(A,B)
0	0	1
0	1	0
1	0	0
1	1	1

Y7 = AB + A'B'

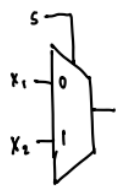
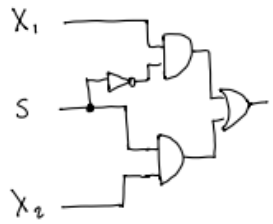
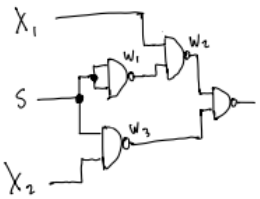
1.8 3-input NAND gate

Draw 3-input NAND	Truth Table and Simplest Sum of Products Equation	NOT-AND-OR Equivalent Circuit	all-NAND Equivalent Circuit
			

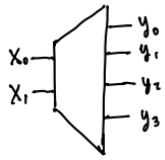
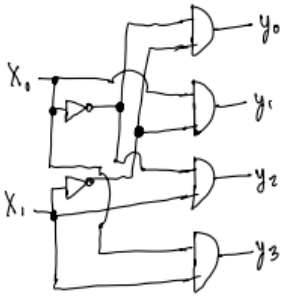
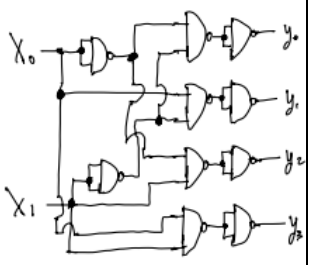
X_2	X_1	X_0	NAND3
0	0	0	1
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	0

Y8 = A' + B' + C'

1.9- 2 to 1 Encoder or Multiplexer (Mux)

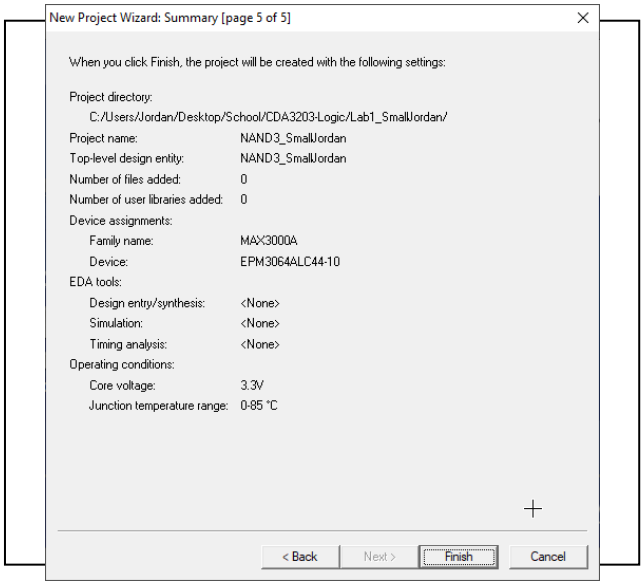
<table>
<tr><th>Draw 2to1 Mux</th><th>Truth Table and Simplest Sum of Products Equation</th><th>NOT-AND-OR Equivalent Circuit</th><th>all-NAND Equivalent Circuit</th></tr>
<tr>
<td> 
</td>
<td>
<table>
<tr><th>s</th><th>X_1</th><th>X_2</th><th>Mux</th></tr>
<tr><td>0</td><td>0</td><td>0</td><td>0</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>0</td></tr>
<tr><td>0</td><td>1</td><td>0</td><td>1</td></tr>
<tr><td>0</td><td>1</td><td>1</td><td>1</td></tr>
<tr><td>1</td><td>0</td><td>0</td><td>0</td></tr>
<tr><td>1</td><td>0</td><td>1</td><td>1</td></tr>
<tr><td>1</td><td>1</td><td>0</td><td>0</td></tr>
<tr><td>1</td><td>1</td><td>1</td><td>1</td></tr>
</table>
Y8 = S'X1 + SX2
</td>
<td> 
</td>
<td> 
</td>
</tr>
</table>

1.10 - 2 to 4 decoder or Demultiplexer (DMux)

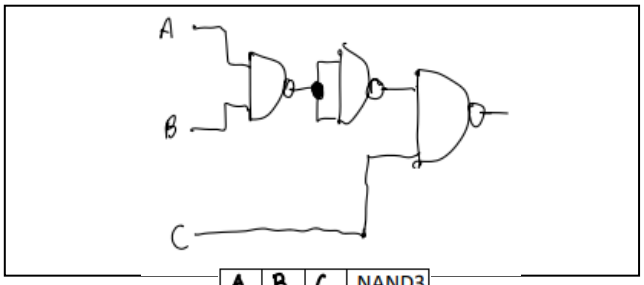
<table>
<tr><th>Draw 2to4 DMux</th><th>Truth Table and Simplest Sum of Products Equation</th><th>NOT-AND-OR Equivalent Circuit</th><th>all-NAND Equivalent Circuit</th></tr>
<tr>
<td> 
</td>
<td>
<table>
<tr><th>X_1</th><th>X_0</th><th>Y_3</th><th>Y_2</th><th>Y_1</th><th>Y_0</th></tr>
<tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>1</td></tr>
<tr><td>0</td><td>1</td><td>0</td><td>0</td><td>1</td><td>0</td></tr>
<tr><td>1</td><td>0</td><td>0</td><td>1</td><td>0</td><td>0</td></tr>
<tr><td>1</td><td>1</td><td>1</td><td>0</td><td>0</td><td>0</td></tr>
</table>
Y3= X1X0
Y2= X1X0'
Y1= X1'X0
Y0=X1'X0'
</td>
<td> 
</td>
<td> 
</td>
</tr>
</table>

2.1 Create a new project in Altera Quartus using VHDL of a **3-input NAND** circuit call it NAND3_YourName, using all-NAND gates

Project Wizard



3-input NAND Diagram

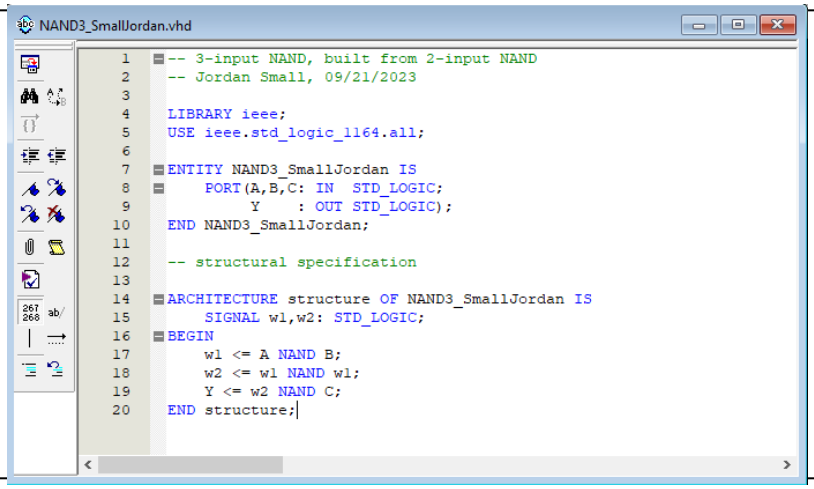


Truth Table

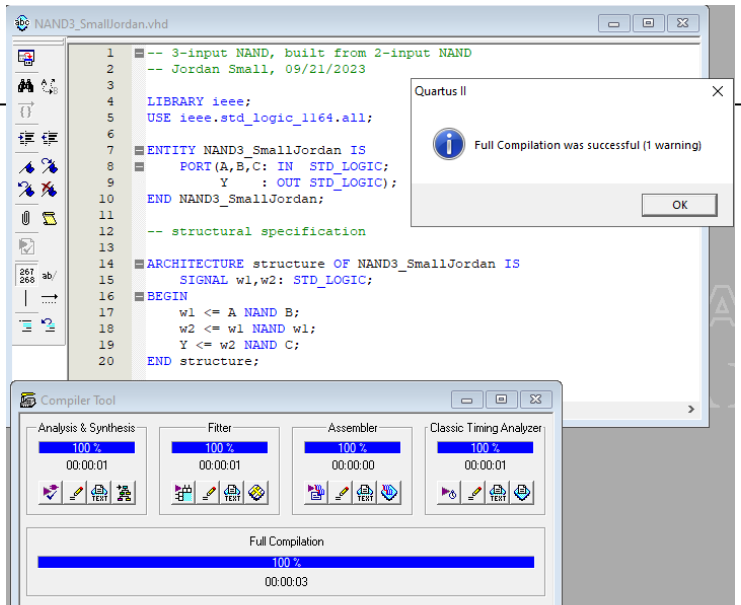
A	B	C	NAND3
0	0	0	1
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	0

Y8= A'+B'+C'

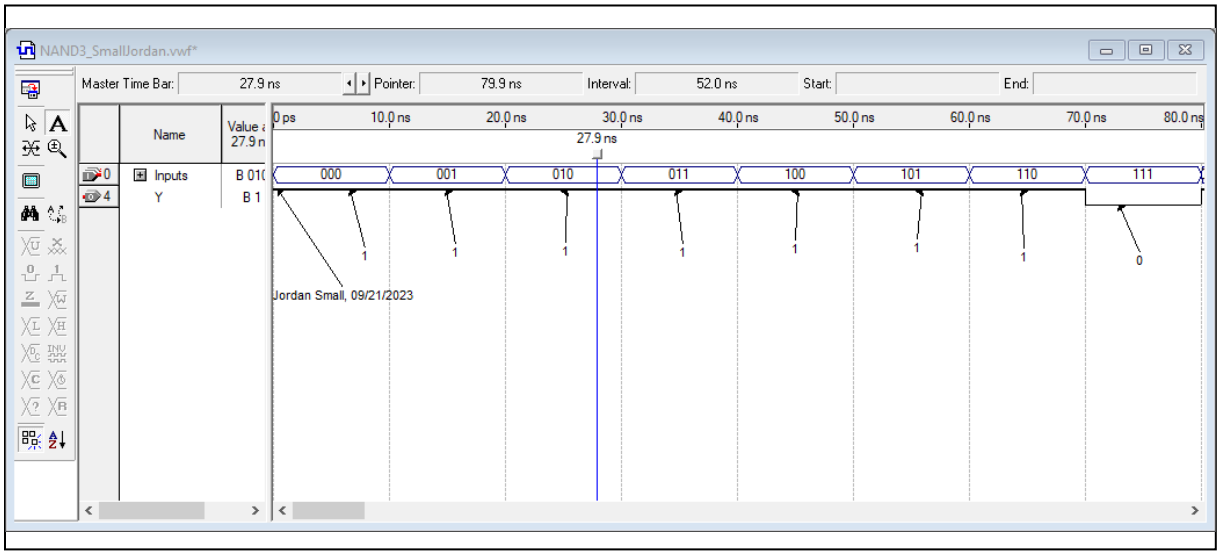
VHDL code



Successful Compilation

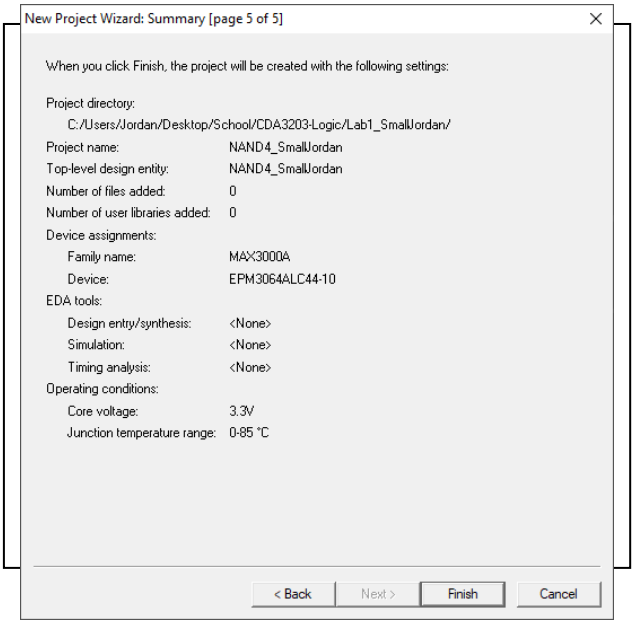


Timing Diagram



2.2 Create a new project in Altera Quartus using VHDL of a **4-input NAND** circuit, call it NAND4_YourName, **using all-NAND gates**

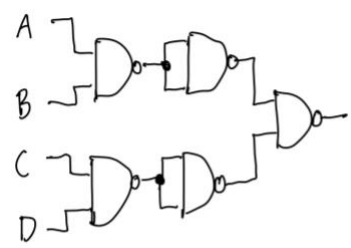
Project Wizard



Truth Table

A	B	C	D	Y
0	0	0	0	1
0	0	0	1	1
0	0	1	0	1
0	0	1	1	1
0	1	0	0	1
0	1	0	1	1
0	1	1	0	1
0	1	1	1	1
1	0	0	0	1
1	0	0	1	1
1	0	1	0	1
1	0	1	1	1
1	1	0	0	1
1	1	0	1	1
1	1	1	0	1
1	1	1	1	0

4-input NAND Diagram

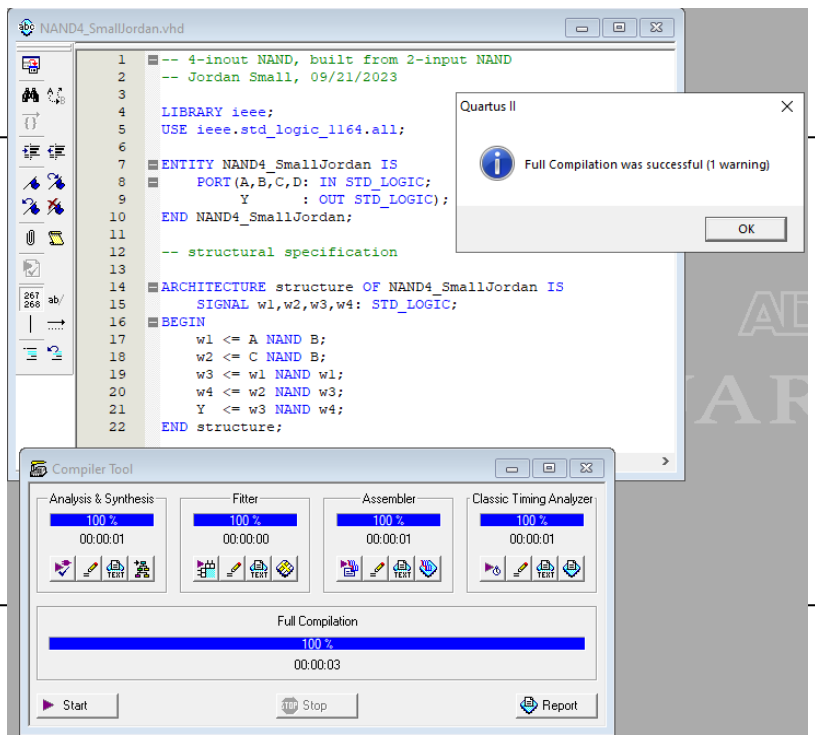


VHDL code

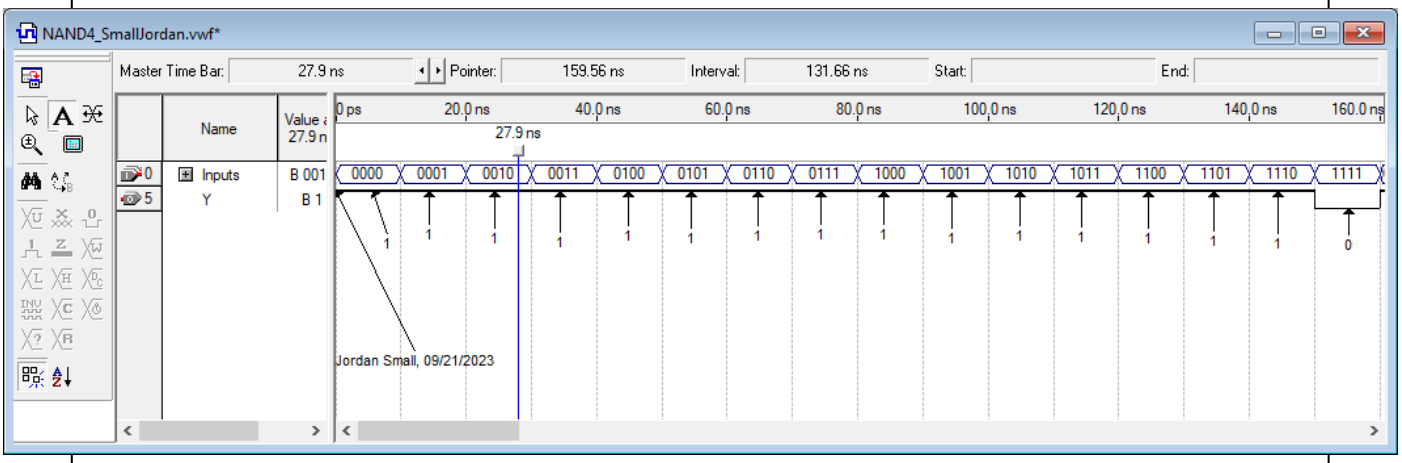
NAND4_SmallJordan.vhd

```vhdl
-- 4-inout NAND, built from 2-input NAND
-- Jordan Small, 09/21/2023

LIBRARY ieee;
USE ieee.std_logic_1164.all;

ENTITY NAND4_SmallJordan IS
    PORT(A,B,C,D: IN STD_LOGIC;
         Y      : OUT STD_LOGIC);
END NAND4_SmallJordan;

-- structural specification

ARCHITECTURE structure OF NAND4_SmallJordan IS
    SIGNAL w1,w2,w3,w4: STD_LOGIC;
BEGIN
    w1 <= A NAND B;
    w2 <= C NAND B;
    w3 <= w1 NAND w1;
    w4 <= w2 NAND w3;
    Y  <= w3 NAND w4;
END structure;
```

Successful Compilation

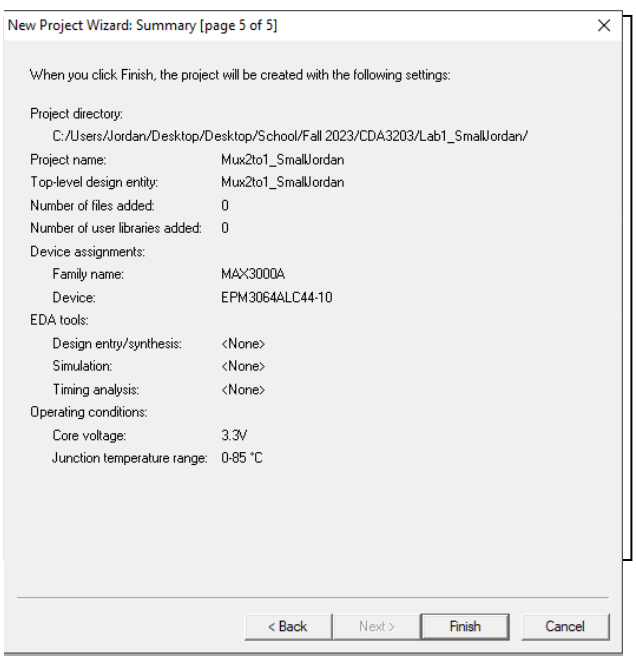


Timing Diagram



2.3 Create a new project in Altera Quartus using VHDL of a **2 to 1 Multiplexer/Encoder (Mux)** circuit, call it Mux2to1_YourName, **using all-NAND gates**

Project Wizard

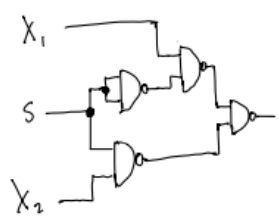


Truth Table

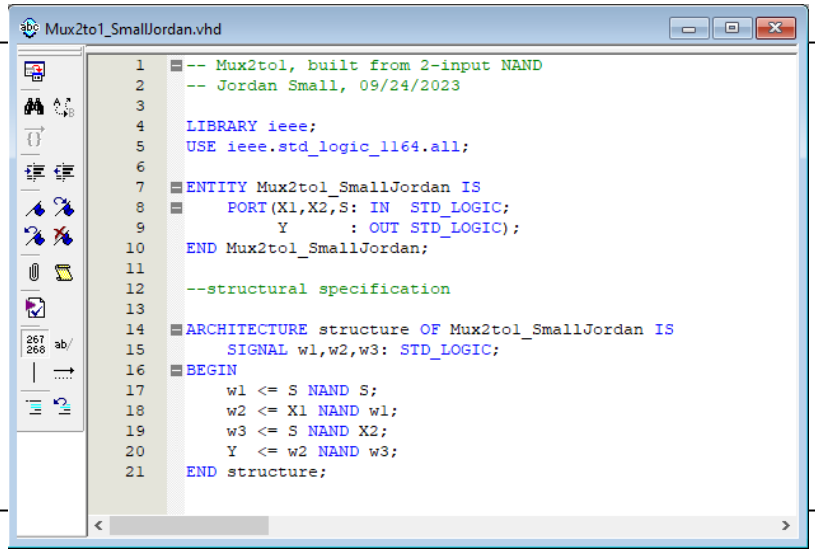
S	X_1	X_2	Mux
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	1

$Y9 = S'X_1 + SX_2$

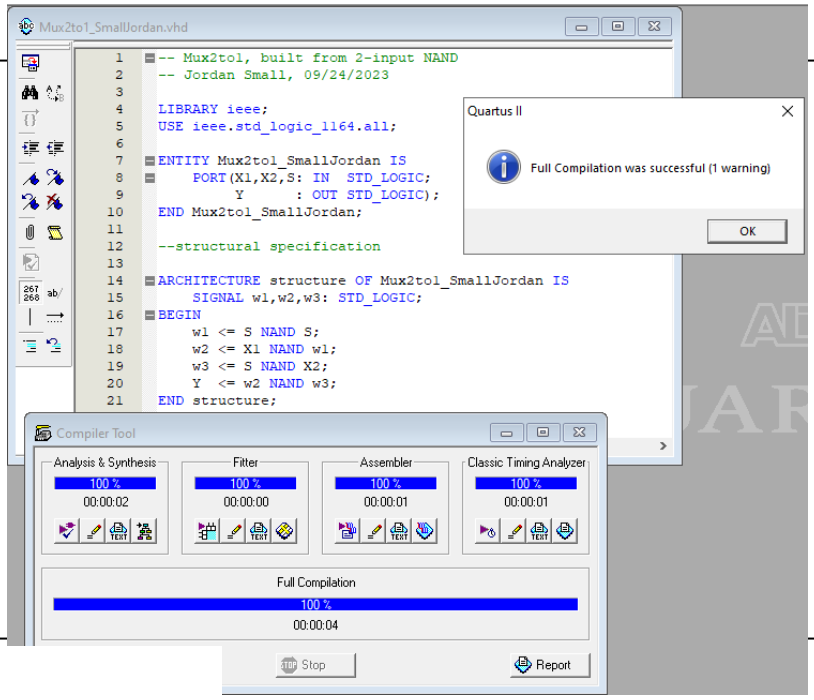
Mux2to1 Diagram



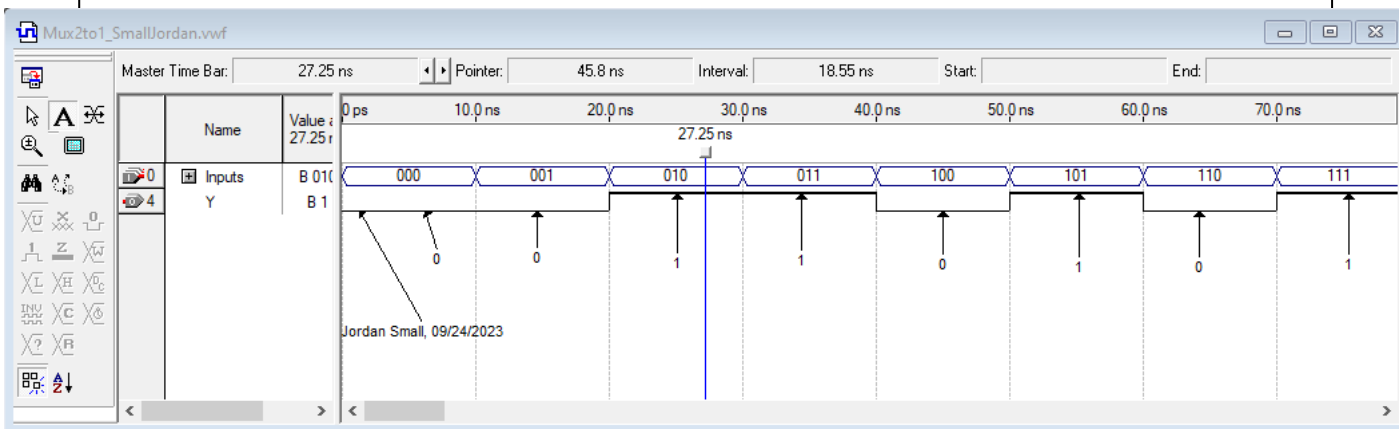
VHDL code



Successful Compilation

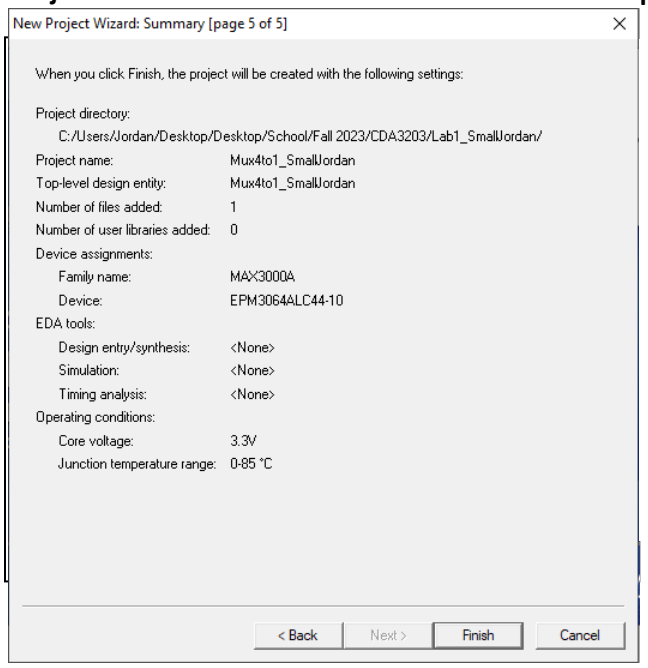


Timing Diagram

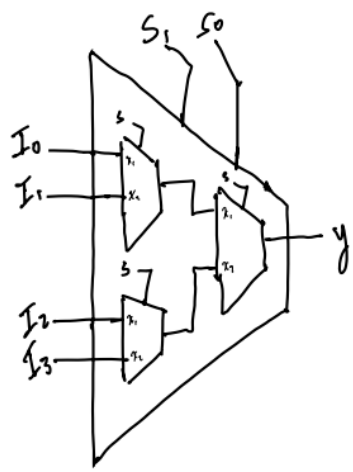


2.4. Create a new project in Altera Quartus using VHDL of a **4-to-1 Mux** circuit, call it Mux4to1_YourName, **using only Mux2to1 components**.

Project Wizard



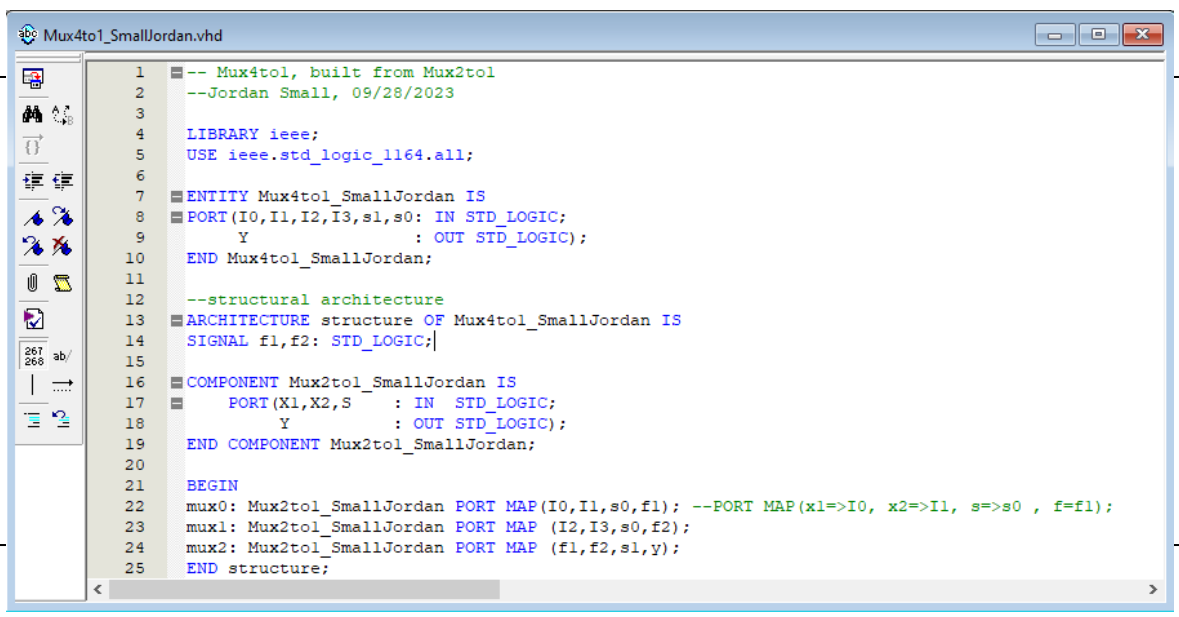
Mux4to1 Diagram



Truth Table

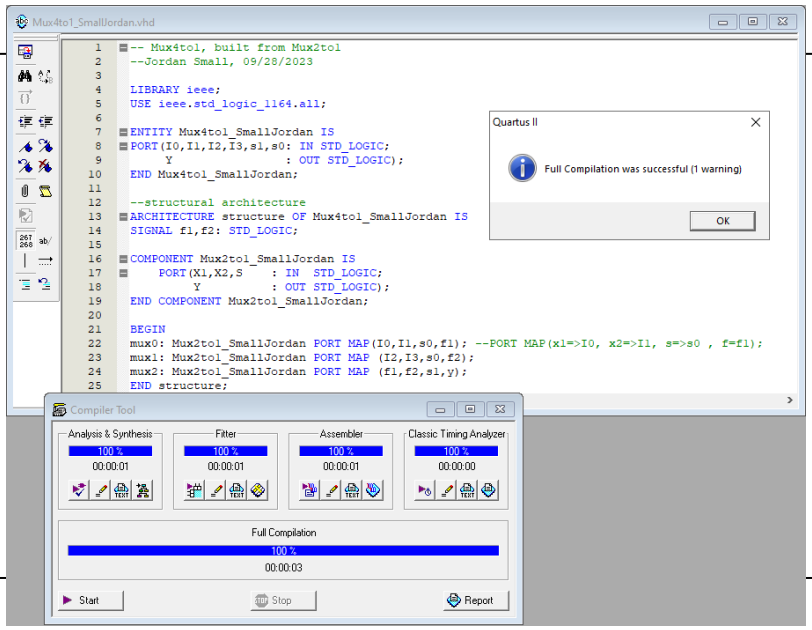
S_1	S_0	y
0	0	I_0
0	1	I_1
1	0	I_2
1	1	I_3

VHDL code

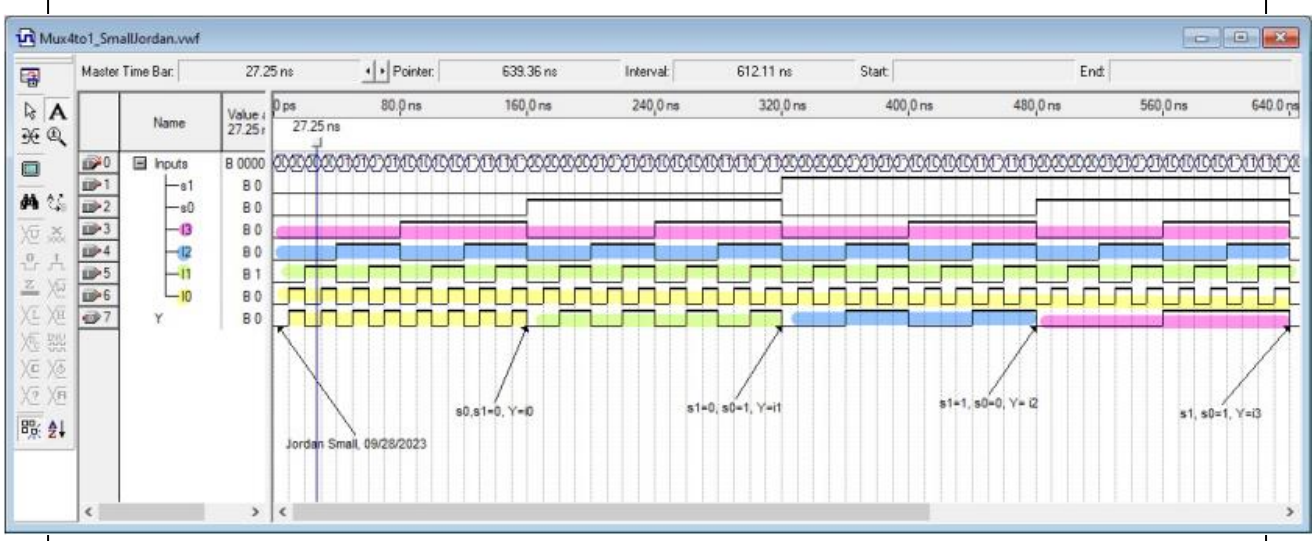



```
-- Mux4to1, built from Mux2to1
--Jordan Small, 09/28/2023

LIBRARY ieee;
USE ieee.std_logic_1164.all;

ENTITY Mux4to1_SmallJordan IS
PORT(I0,I1,I2,I3,s1,s0: IN STD_LOGIC;
     Y                  : OUT STD_LOGIC);
END Mux4to1_SmallJordan;

--structural architecture
ARCHITECTURE structure OF Mux4to1_SmallJordan IS
SIGNAL f1,f2: STD_LOGIC;

COMPONENT Mux2to1_SmallJordan IS
    PORT(X1,X2,S    : IN  STD_LOGIC;
         Y          : OUT STD_LOGIC);
END COMPONENT Mux2to1_SmallJordan;

BEGIN
mux0: Mux2to1_SmallJordan PORT MAP(I0,I1,s0,f1); --PORT MAP(x1=>I0, x2=>I1, s=>s0 , f=f1);
mux1: Mux2to1_SmallJordan PORT MAP (I2,I3,s0,f2);
mux2: Mux2to1_SmallJordan PORT MAP (f1,f2,s1,y);
END structure;
```

Successful Compilation

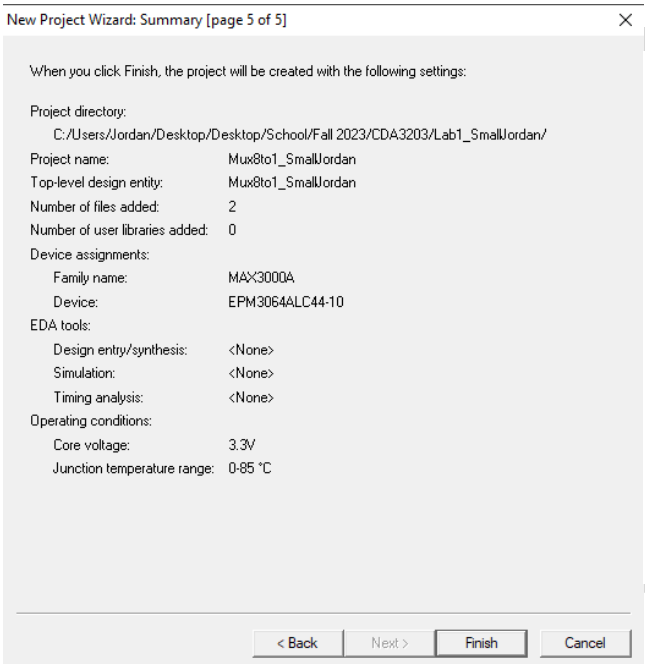


Timing Diagram

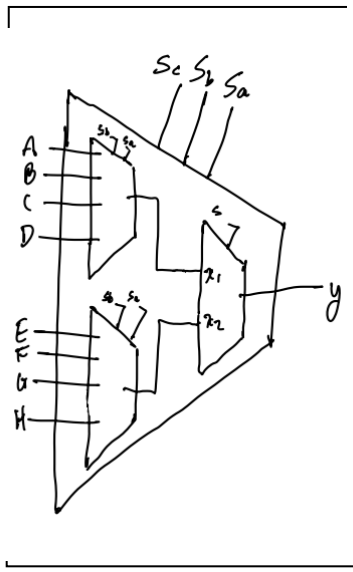


2.5 Create a new project in Altera Quartus using VHDL of a **8-to-1 Mux**, call it Mux8to1_YourName, using **only Mux4to1 and/or Mux2to1components**

Project Wizard



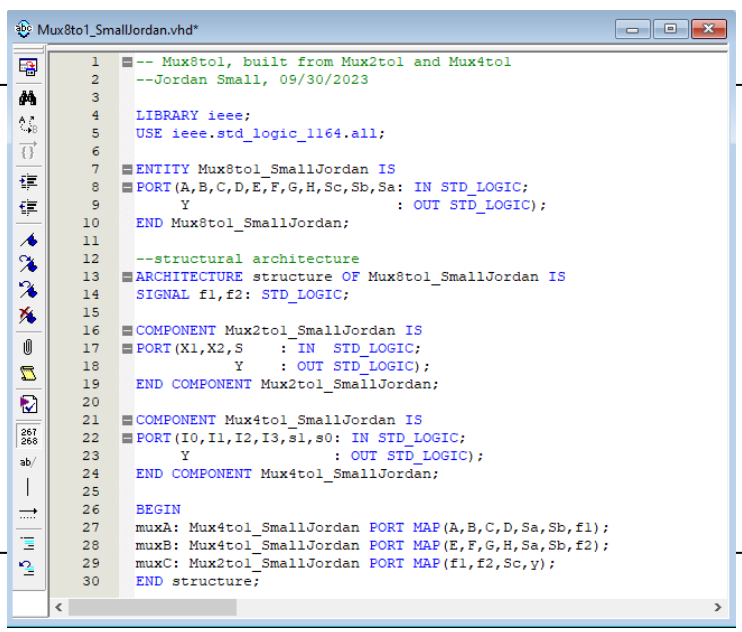
Mux8to1 Diagram



Truth Table

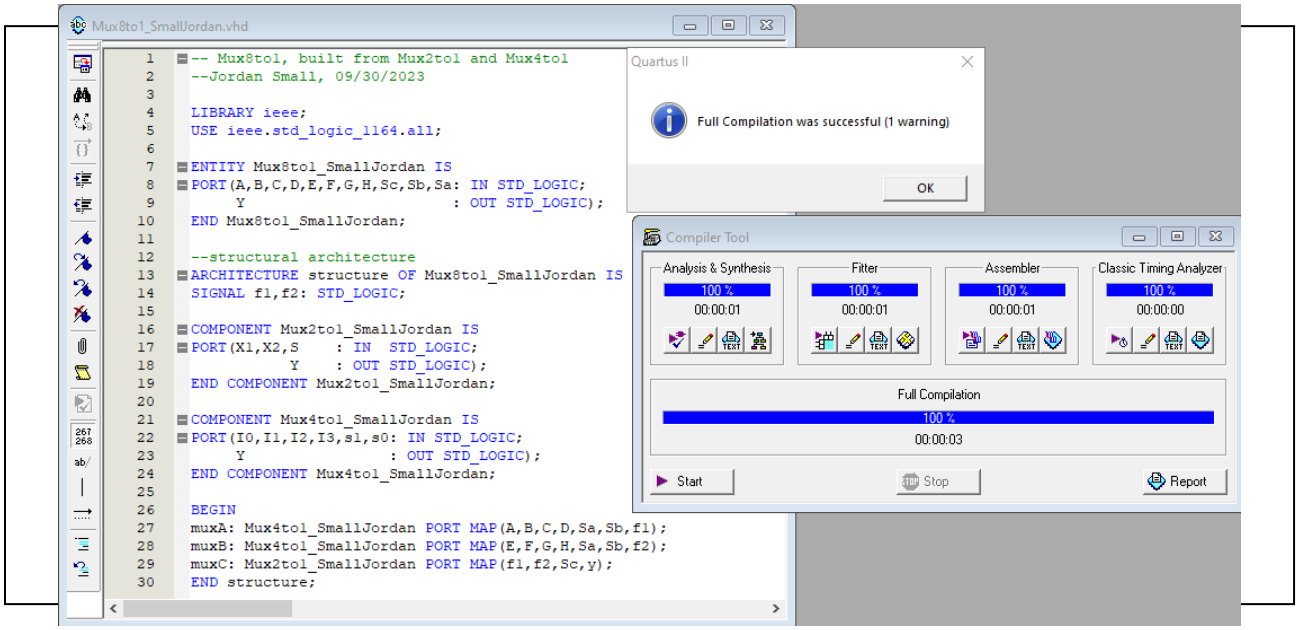
Sc	Sb	Sa	y
0	0	0	A
0	0	1	B
0	1	0	C
0	1	1	D
1	0	0	E
1	0	1	F
1	1	0	G
1	1	1	H

VHDL code

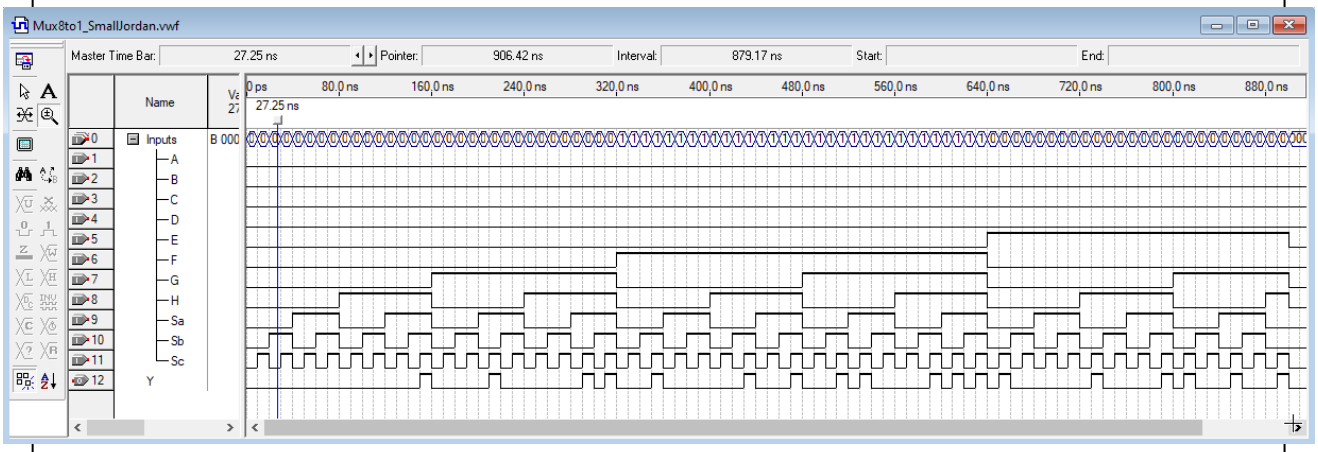



```vhdl
-- Mux8to1, built from Mux2to1 and Mux4to1
--Jordan Small, 09/30/2023

LIBRARY ieee;
USE ieee.std_logic_1164.all;

ENTITY Mux8to1_SmallJordan IS
PORT(A,B,C,D,E,F,G,H,Sc,Sb,Sa: IN STD_LOGIC;
     Y                         : OUT STD_LOGIC);
END Mux8to1_SmallJordan;

--structural architecture
ARCHITECTURE structure OF Mux8to1_SmallJordan IS
SIGNAL f1,f2: STD_LOGIC;

COMPONENT Mux2to1_SmallJordan IS
PORT(X1,X2,S    : IN  STD_LOGIC;
         Y      : OUT STD_LOGIC);
END COMPONENT Mux2to1_SmallJordan;

COMPONENT Mux4to1_SmallJordan IS
PORT(I0,I1,I2,I3,s1,s0: IN STD_LOGIC;
     Y                  : OUT STD_LOGIC);
END COMPONENT Mux4to1_SmallJordan;

BEGIN
muxA: Mux4to1_SmallJordan PORT MAP(A,B,C,D,Sa,Sb,f1);
muxB: Mux4to1_SmallJordan PORT MAP(E,F,G,H,Sa,Sb,f2);
muxC: Mux2to1_SmallJordan PORT MAP(f1,f2,Sc,y);
END structure;
```

Successful Compilation

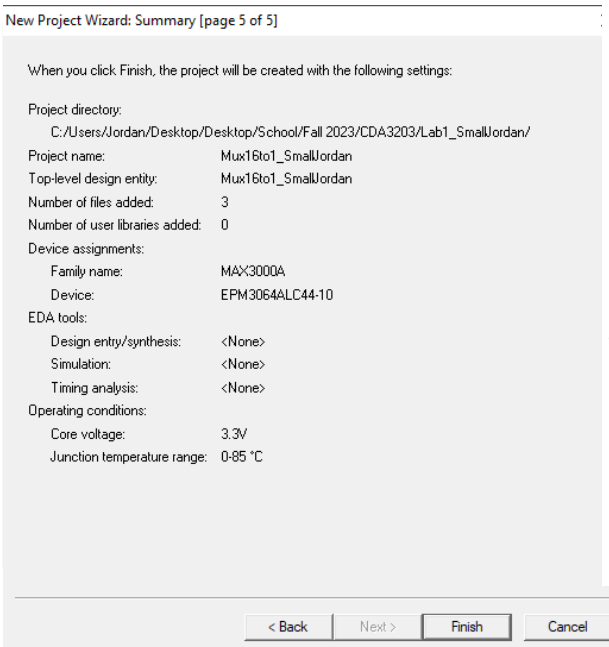


Timing Diagram

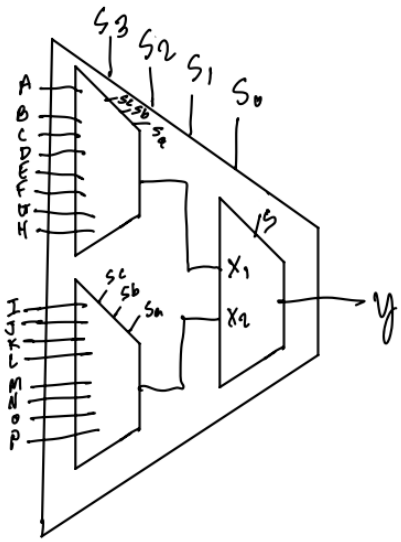


2.6 Create a new project in Altera Quartus using VHDL of a **16-to-1 Mux**, call it Mux8to1_YourName, **using only Mux8to1, Mux4to1 and/or Mux2to1 components**

Project Wizard



Mux16to1 Diagram



Truth Table

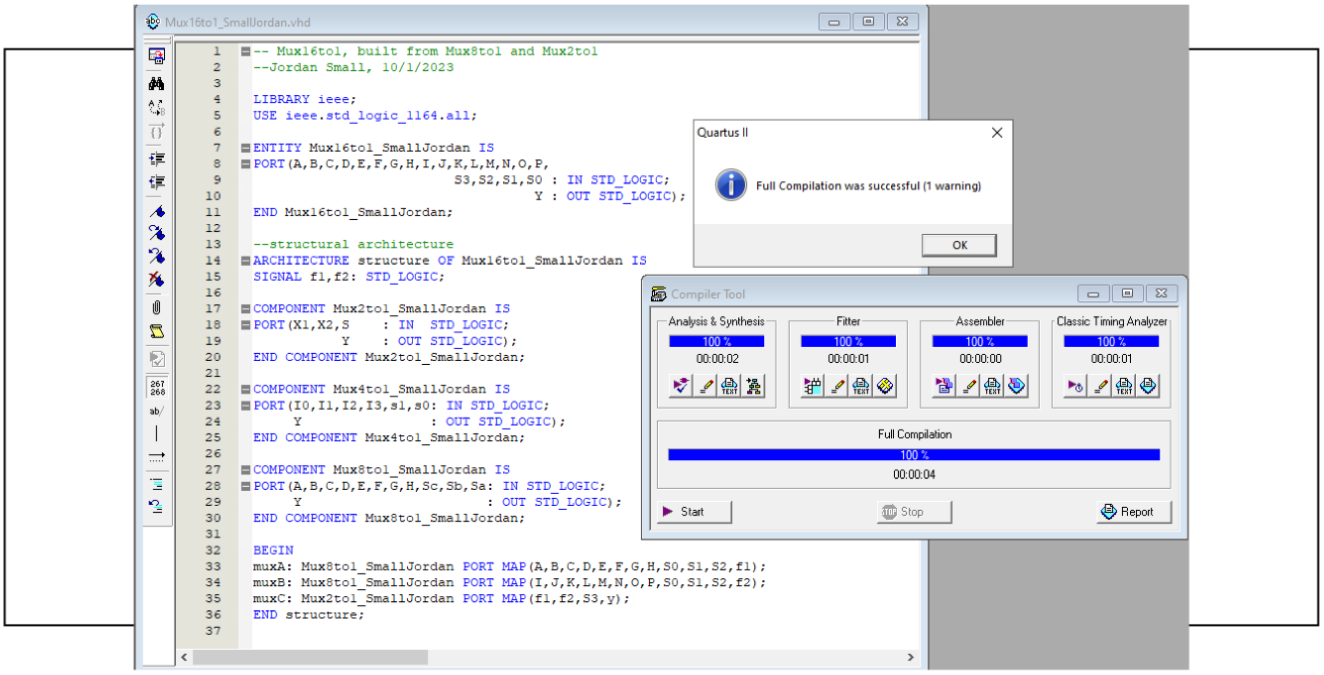
S_3	S_2	S_1	S_0	y
0	0	0	0	A
0	0	0	1	B
0	0	1	0	C
0	0	1	1	D
0	1	0	0	E
0	1	0	1	F
0	1	1	0	G
0	1	1	1	H
1	0	0	0	I
1	0	0	1	J
1	0	1	0	K
1	0	1	1	L
1	1	0	0	M
1	1	0	1	N
1	1	1	0	O
1	1	1	1	P

VHDL code

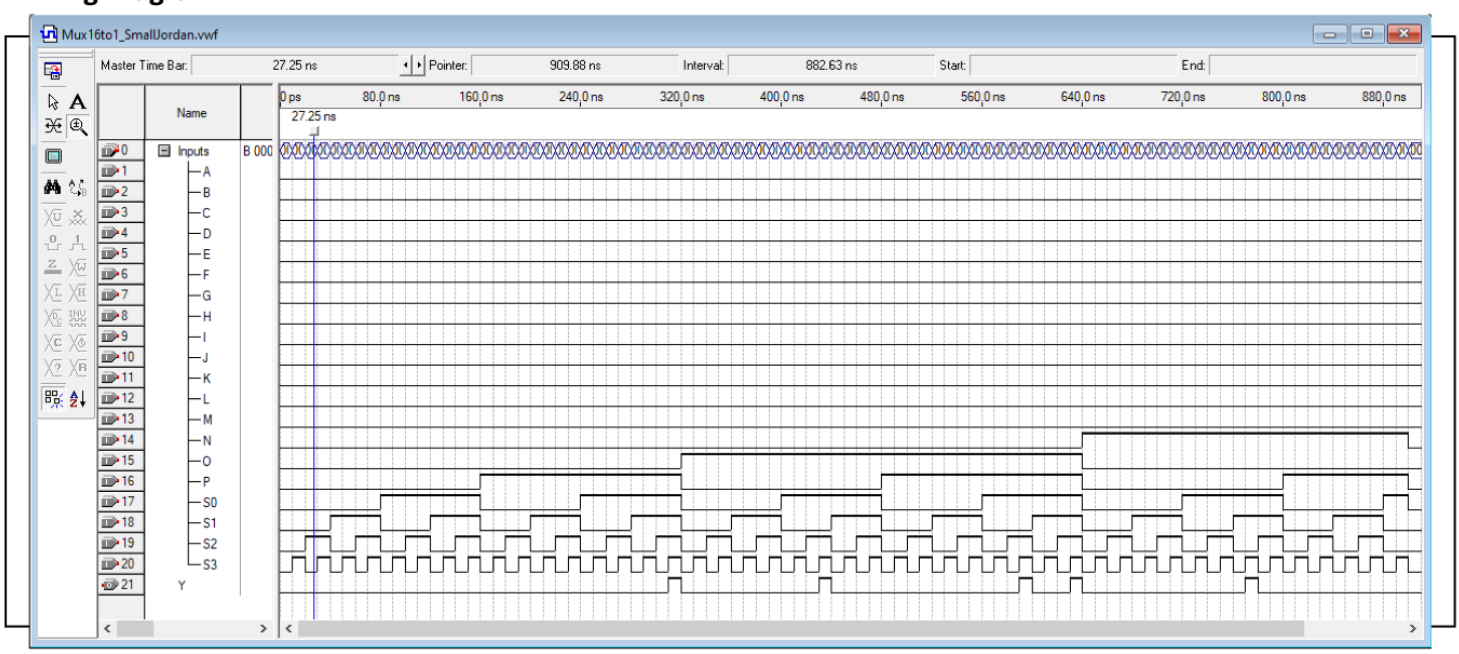
Mux16to1_SmallJordan.vhd

```vhdl
-- Mux16to1, built from Mux8to1 and Mux2to1
--Jordan Small, 10/1/2023

LIBRARY ieee;
USE ieee.std_logic_1164.all;

ENTITY Mux16to1_SmallJordan IS
PORT(A,B,C,D,E,F,G,H,I,J,K,L,M,N,O,P,
                        S3,S2,S1,S0 : IN STD_LOGIC;
                                  Y : OUT STD_LOGIC);
END Mux16to1_SmallJordan;

--structural architecture
ARCHITECTURE structure OF Mux16to1_SmallJordan IS
SIGNAL f1,f2: STD_LOGIC;

COMPONENT Mux2to1_SmallJordan IS
PORT(X1,X2,S    : IN  STD_LOGIC;
           Y    : OUT STD_LOGIC);
END COMPONENT Mux2to1_SmallJordan;

COMPONENT Mux4to1_SmallJordan IS
PORT(I0,I1,I2,I3,s1,s0: IN STD_LOGIC;
      Y                 : OUT STD_LOGIC);
END COMPONENT Mux4to1_SmallJordan;

COMPONENT Mux8to1_SmallJordan IS
PORT(A,B,C,D,E,F,G,H,Sc,Sb,Sa: IN STD_LOGIC;
      Y                        : OUT STD_LOGIC);
END COMPONENT Mux8to1_SmallJordan;

BEGIN
muxA: Mux8to1_SmallJordan PORT MAP(A,B,C,D,E,F,G,H,S0,S1,S2,f1);
muxB: Mux8to1_SmallJordan PORT MAP(I,J,K,L,M,N,O,P,S0,S1,S2,f2);
muxC: Mux2to1_SmallJordan PORT MAP(f1,f2,S3,y);
END structure;
```

Successful Compilation



Timing Diagram



This project was done by: Jordan Small

following the tutorial videos created by Dr. Petrie and material in the class textbook,
with no other outside resources or help with the following exceptions:
_x_ Received help from Teaching Assistant: Harry (Timing diagrams for Mux8to1 and Mux16to1; he said no annotation was necessary for these)
___ Found the following material on the internet: show URL
___ Other: